RSA分散復号 (代数系アルゴリズムと言語および計算理論)

AUTHOR(S):

足立, 智子; 光野, 繁治

CITATION:

足立, 智子 ...[et al]. RSA分散復号 (代数系アルゴリズムと言語および計算理論). 数理解析研究所講究録 2009, 1655: 111-123

ISSUE DATE:

2009-06

URL:

http://hdl.handle.net/2433/140851

数理解析研究所講究録
第1655巻 2009年 111-123

RSA分散復号

東邦大学 理学部 情報科学科
足立 智子, 光野 繁治
Toho University, Department of Information Science
Tomoko Adachi, Shigeharu Mitsuno

1. はじめに

一般に公開鍵暗号系では、各ユーザが公開鍵と秘密鍵を持ち、公開鍵を公開している。その公開鍵を用いて他のユーザは暗号化を行い、自分自身宛の暗号文は秘密鍵を用いて復号化を行う。このようにして通信を行う際に一つのメッセージに注目すると、一人の送信者に対して受信者も一人となる。しかし、必ずしも一人対一人とは限らず、受信者が複数いる場合も存在する。その場合、鍵を分散させて復号化を行う必要がある。RSA分散復号[5]とは、そのような分散復号環境においてRSA暗号[6]を実現する方法である。また、分散復号環境においてElGamal暗号[2]を実現するElGamal分散復号[3]もある。本稿では、これらについて紹介する。

2. RSA分散復号

通常、RSA暗号のような公開鍵暗号系では、受信者と送信者は一人ずつである。しかし、必ずしも一人ずつとは限らず、複数の受信者からなる受信者グループの場合も存在する。このような場合における通信は、受信者が一人の場合と同様な流れで通信を行うことができる。グループとして公開鍵と秘密鍵を作成し、送信者はその公開鍵を用いメッセージを暗号化し、受信者グループに送る。受け取った暗号文を秘密鍵を用いて復号化を行い、メッセージを得ることができる。しかし、問題点も存在する。それは、秘密鍵をどのようにして保管するかという問題である。秘密鍵が一つの場合では、保管している人だけが勝手に復号化を行えてしまう。全員が秘密鍵を保管する場合では、全員が勝手に復号化を行えてしまう。また、秘密鍵の数が増えるため、秘密鍵が盗聴されるというリスクも増加してしまう。これらの問題を解決するのが、秘密鍵を分散させて各受信者が管理するというRSA分散復号である。まず、RSA分散復号の基盤となるRSA暗号、構成するために必要な知識として秘密分散法,マルチパー

ティプロトコル,検証についての説明を行い、RSA 分散復号の構成について説明を行っていく。

2.1 RSA 暗号

RSA 暗号とは、Rivest, Shamir, Adleman によって考案された、素数の数学的性質である素因数分解の求解困難性に基づいた暗号である。大きく分けて鍵の生成・暗号化・復号化の三つの手順から構成されている。まず通信を行う前に準備として、受信者は公開鍵と秘密鍵を作成し公開鍵を公開する。次に送信者は、公開されている公開鍵を用いてメッセージを暗号化し、暗号文を受信者へ送る。最後に受信者は、自分が保管している秘密鍵を用いて暗号文の復号化を行い、メッセージを手に入れる。以上が全体の流れとなる。次に、鍵の生成・暗号化・復号化それぞれの手順について説明を行う。

2.1.1 鍵の生成

通信を行う前に、受信者側が鍵の生成を行う。

アルゴリズム 2.1.1　RSA 暗号における鍵の生成

Step 1: 二つの素数 p, q を生成する。

Step 2: $n=pq$, $\lambda(n)=\mathrm{LCM}(p-1, q-1)$を計算する。

Step 3: $\mathrm{GCD}(e, \lambda(n))=1$ となる e を定める。このような e は、複数存在する。

Step 4: $ed \equiv 1 \pmod{\lambda(n)}$となる d を計算する。このような d は、e に対して必ず 1 つ存在する。

以上のようにして求めた e, n を公開鍵として公開し、d を秘密鍵として本人だけが知るように保管する。鍵の生成時の注意として、素数 p, q は Step 2 を終えた時点で破棄する必要がある。これは、p, q が知られてしまうと結果的に秘密鍵 d が知られてしまうためである。また、n が素因数分解されても p, q が知られてしまうため、p と q はそれぞれ 150 桁程度必要となる。

2.1.2 暗号化

送信者が受信者にメッセージ m を送るとする。送信者は受信者が公開している二つの公開鍵 n, e を探し、それらを用いて次のように暗号化を行う。

$$c \equiv m^e \pmod{n}$$

得られた暗号文 c を受信者へ送る。

2.1.3 復号化

受け取った暗号文 c を、自分で保管していた秘密鍵 d を用いて次のように復号化を行う。

$$m \equiv c^d \pmod{n}$$

このようにしてメッセージ m を復元する。なぜこのように復号化が行えるかというと、まず暗号化の式を両辺 d 乗する。

$$c^d \equiv m^{ed} \pmod{n}$$

ここで ed に注目すると、鍵の生成時にλ(n)を法として 1 と合同としたので、ある定数 k を用いて

$$c^d \equiv m^{k\lambda(n)+1} \pmod{n}$$

と変形できる。次に、累乗の部分に注目し式変形を行ってくと

$$c^d \equiv (m^{\lambda(n)})^k m \pmod{n}$$

となる。最後に、オイラーの定理を用いることで $m^{\lambda(n)}$ は 1 となるので、結果として

$$c^d \equiv m \pmod{n}$$

が得られる。このようにして秘密鍵 d を用いて復号化が可能となる。

2.2 秘密分散法

秘密分散法とは、秘密情報を持つディーラと複数の参加者間におけるプロトコルである。分散フェーズと復号フェーズから成る。まず、ディーラが秘密情報を複数の分散情報に分割し、それぞれの分散情報を参加者に秘密裏に送る。次に、参加者達は自分達の持つ分散情報を集めることで、元の秘密情報を復元することができる。元の秘密情報を復元するために必要な分散情報が、分割した数と同じとなる満場一致法と、ある決められた数以上となるしきい値法がある。

2.2.1 満場一致法

満場一致法とは、秘密分散法において、全ての分散情報を集めることで初めて秘密情報を復元できる方法である。まず、ディーラが秘密情報から参加者人数分の分散情報を作成し、それぞれの分散情報を各参加者へ秘密裏に送る。次に、全参加者が分散情報を公開し、秘密情報の復元を行う。仮に盗聴者が誰かから分散情報を盗んだとしても、一つの分散情報からでは元の秘密を得ることはできません。全ての分散情報を盗もうとしても、何個あるのかも分からない分散情報を全て集めることはできない。また、参加者内に単独、あるいは共謀して不正を行うものが現れたとしても、元の秘密情報を正しく復元することはできない。

秘密分散法を他の暗号プロトコルに応用する場合、満場一致法では不都合な場合がある。例えば、なんらかの理由で参加者の一人が分散情報を紛失した場合、全ての分散情報を集めることができないので、元の秘密情報を復元することはできなくなる。このようなことを回避するために考案された方法がしきい値法である。

2.2.2 (t, N)しきい値法

しきい値法の一つである(t, N)しきい値法とは、秘密分散法において、N 人の参加者のうち t 人以上により秘密情報を復元できる方法である。これまでと同様にディーラが分散情報を配り、t 人以上の参加者が分散情報を公開することで秘密情報を復元することができる。言い換えれば、N−t 人の参加者がなんらかの理由で分散情報を紛失したとしても、元の秘密情報を復元することができるということである。この方法の実現例として、シャミアにより提案されたシャミアのしきい値法というものがある。

2.2.3 シャミアのしきい値法

シャミアのしきい値法とは、シャミアによって提案された多項式を利用した方法である。今、ある 2 次関数があったとする。この関数上の 3 点以上の点と 2 次関数であることが分かれば、どのような 2 次間数であるかを求めることはできる。しかし、3 点未満の点からでは求めることができない。この性質を利用している。つまり、N 個中 t 個以上の分散情報により秘密情報を復元するには t−1 次の関数が必要となる。

ディーラが N+1 より大きな素数 p に対して、$s \in F_p$ である秘密情報 s を持っているとする。まずディーラは、乱数 $a_1, \cdots, a_{t-1}$ を F_p からランダムに選び、s が

切片となるt−1次の秘密の多項式

$$f(x) = s + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1} (\mathrm{mod}\ p)$$

を作成する。次に、N個のf(x)上の点を求め、それぞれを各参加者へ送る。復元ではまず、t人以上の参加者が分散情報を公開し、それらの点を通る多項式f(x)を求める。次に、求めたf(x)にx=0を代入することで、切片である秘密情報sを得ることができる。

2.3 マルチパーティプロトコル

マルチパーティプロトコルとは、複数の参加者$A_i (i > 0)$がそれぞれの秘密情報s_iを持つ状況下において、秘密情報を秘匿したまま、秘密から計算される関数$f(s_1, s_2, \cdots)$を計算するプロトコルである。例として、秘密情報の和

$$f(s_1, s_2, \cdots) = s_1 + s_2 \cdots$$

を求めるマルチパーティプロトコルを考えてみる。例えば、全参加者から信頼されたセンタが存在すると仮定した場合を考えてみる。その場合、各参加者A_iがセンタに秘密情報s_iを秘密裏に送り、センタは送られた秘密情報から参加者が求めたい計算を行い、その計算した値を各参加者へ伝える。これにより、互いに秘密情報を知られることなく結果を得ることができる。しかし、このようなセンタを仮定せずに同様のことを行うプロトコルがマルチパーティプロトコルである。マルチパーティプロトコルは、秘密分散法を利用することで実現可能である。ただし条件として、任意の2人の参加者間には秘密通信路が存在し、公開掲示板によって情報の公開が可能とする。

アルゴリズム 2.3.1 秘密情報の和を求めるマルチパーティプロトコル

Step 1: N人の参加者A_N、i番目の参加者A_iの秘密情報をs_iとする。また、参加者間で秘密情報の和より十分に大きな素数pを事前に共有しておく。

Step 2: シャミアのしきい値法と同様に各参加者A_iは、$f_i(0) = s_i$を満たすt−1次のF_p上の多項式f_iを作成する。その多項式上の点$v_{i,j} = f_i(j)$を計算し、A_jへ秘密裏に送る。

Step 3: Step1により、各参加者A_jは全参加者の多項式上の点$v_{1,j}, \cdots v_{N,j}$を得る。A_jは、$v_j = v_{1,j} + \cdots + v_{N,j} (\mathrm{mod}\ p)$を計算する。

Step 4: 各A_jは求めたv_jを公開する。シャミアのしきい値法と同様に公開された全てのv_jから、$f(j) = v_j$を満たすt−1次の多項式を求める。f(0)を求めることでsを得る。ここで、sがs_iの和になっている。

シャミアのしきい値法の線形性により、上記のような秘密情報の和だけでなく、任意の秘密の線形結合を計算するマルチパーティプロトコルも同様に作ることができる。一般のマルチパーティプロトコルは次のようになる。

アルゴリズム 2.3.2 一般のマルチパーティプロトコル

Step 1: 計算に必要な秘密を持っている参加者が(t, N)しきい値法の分散フェーズを行って秘密を分散させる。

Step 2: 各参加者がそれぞれ、または協力して入力値のシェアから出力値のシェアを計算し、全ての参加者間に分散共有させる。

Step 3: 各参加者が出力値のシェアを明かし、(t, N)しきい値法の復元フェーズを行い、秘密を復元する。

ポイントとしては、Step1 では各参加者はディーラとして、Step3 では参加者として行動しているということである。全ての参加者が正しく計算を行えば、このプロトコルは常に求めたい関数を正しく計算し、結果以外の情報は漏らさない。しかし、参加者のうち一人でも不正を行う者がいると、正しい結果を得ることはできない。そのため、不正を行った者がいるかをチェックする検証という仕組みが必要となる。

2.4 検証

第 1.3 節で述べたように、あるプロトコルによって定められた方法で計算を行えば、そのプロトコルは正常に作動する。しかし、不正を行うものが一人でも存在すると、プロトコルは正常に作動せず、得たいものが得られなくなってしまう。これを防ぐために、不正を行った者がいるかをチェックする仕組みが必要となる。その仕組みが検証である。例えば、第 1.3 節で説明した場合では二つの検証が必要となる。一つ目は、各参加者がディーラとして正しく秘密を計算しているかどうかである。二つ目は、各参加者が受け取った値から正しく計算を行い、その結果を正しく公開しているかどうかである。このような検証するための仕組みの構成方法としては、最もシンプルなゼロ知識証明を用いる方法や、一方向性関数を利用する方法がある。

2.5 満場一致 RSA 復号

満場一致 RSA 復号とは、秘密鍵を分散して管理している受信者全てによりメ

ッセージを復号化するRSA分散復号である。RSA暗号同様、受信者グループとしての公開鍵を(e, n), 秘密鍵をdとする。このとき、グループのメンバはN人とし、各メンバは

$$d \equiv \sum_{i=1}^{N} d_i \pmod{\lambda(n)}$$

となるような部分秘密鍵d_iをそれぞれ保持するものとする。このグループにメッセージを送る場合、グループが公開している公開鍵(e, n)を用いて暗号化を行い、暗号文をグループへ送る。暗号文を受け取ったグループは、グループのメンバが協力して次のように復号化を行う。

アルゴリズム 2.5.1 満場一意RSA復号における復号化

Step 1: メンバA_iは部分秘密鍵d_iを用いて部分復号情報m_i

$$m_i \equiv c^{d_i} \pmod{n}$$

を計算し、計算した値を公開する。

Step 2: 各メンバは全メンバの部分復号情報から

$$m \equiv \prod_{i=1}^{N} m_i \pmod{n}$$

を計算することでメッセージmを得る。

なぜこのように復号化を行えるかというと、RSA暗号の復号化の式は

$$m \equiv c^{d} \pmod{n}$$

である。ここでdに注目すると、満場一致RSA復号でのdは部分秘密鍵の和だったので

$$m \equiv c^{d_1+d_2+\cdots+d_N} \pmod{n}$$

となる。次に、累乗の部分に注目して式変形を行うと

$$m \equiv c^{d_1} \times c^{d_2} \times \cdots \times c^{d_N} \pmod{n}$$

となる。ここで、cのd_i乗は復号化の式でm_iとなるので

$$m \equiv m_1 \times m_2 \times \cdots \times m_N \pmod{n}$$

となる。この式を書き直すと

$$m \equiv \prod_{i=1}^{N} m_i \pmod{n}$$

となり、結果、各メンバが部分秘密鍵で復号した部分復号情報の積によりメッセージを復元することができる。

ここで述べたのは満場一致RSA復号なので、任意の参加者からメッセージを復元できる(t, N)しきい値RSA復号も存在する。

2.6 (t, N)しきい値RSA復号

前節で述べたように、(t, N)しきい値RSA復号とはシャミアのしきい値法を利用することにより、秘密鍵をN人で分散・管理し、t人以上によりメッセージを復元するRSA分散復号である。秘密鍵dを切片とする多項式fがあり、i番目のメンバが持つ部分秘密鍵d_iを$d_i = f(i)$とする。これは、シャミアのしきい値法における秘密情報を秘密鍵とし、分散情報を部分秘密鍵としている。ラグランジェの補間公式より、λを

$$\lambda_j = \prod_{l \neq j} \frac{-i_l}{i_j - i_l}$$

としたとき、dが

$$d = \sum_{j=1}^{t} \lambda_j d_{ij}$$

となるd_iが部分秘密鍵となる。しかし、これでは問題が生じる。(t, N)しきい値RSA復号の場合、λ(n)は合成数なので$Z_{\lambda(n)}$は体にはならない。また、秘密鍵dは$Z_{\lambda(n)}$の要素となる。一方、シャミアのしきい値法の場合、pは素数なのでZ_pは体F_pとなる。また、秘密情報はF_pの要素でなくてはならない上に、演算は全てF_p上で行わなくてはならない。それゆえ、シャミアのしきい値法をそのままRSA分散復号へ適用して、(t, N)しきい値RSA復号を構成することはできない。シャミアのしきい値法を適用するためには、注意深く部分秘密鍵の作成を行うなどのような条件を複数付け加えることにより、(t, N)しきい値RSA復号の構成が可能となる。

3. ElGamal 分散復号

これまで述べてきたものは、分散復号環境における RSA 暗号の構成法となる RSA 分散復号である。その RSA 分散復号と同様なことが ElGamal 分散復号でもできるので、こちらについても説明を行っていく。

ElGamal 分散復号とは、分散復号環境において ElGamal 暗号を行う方法である。RSA 分散復号同様、秘密分散法,マルチパーティプロトコル,検証を用いることで ElGamal 分散復号を構成できる。これらの知識は第 1 章で説明したので省略し、基盤となる ElGamal 暗号について説明を行い、ElGamal 分散復号の構成法について説明を行っていく。

3.1 ElGamal 暗号

ElGamal 暗号とは、離散対数問題が困難であることを利用している暗号である。大きく分けて鍵の生成・暗号化・復号化の三つの手順から構成されている。まず通信を行う前に準備として、受信者は公開鍵と秘密鍵を作成し公開鍵を公開する。次に送信者は、公開されている公開鍵を用いてメッセージを暗号化し、暗号文を受信者へ送る。最後に受信者は、自分が管理している秘密鍵を用いて暗号文の復号化を行い、メッセージを手に入れる。以上が全体の流れとなる。次に、鍵の生成・暗号化・復号化それぞれの手順について説明を行う。

3.1.1 鍵の生成

通信を行う前に、受信者側が鍵の生成を行う。

アルゴリズム 3.1.1 ElGamal 暗号における鍵の生成

Step 1: 十分に大きな素数 p を生成する。

Step 2: 位数が p－1 となる元 $g \in Z_p^*$ を選ぶ。

Step 3: x を $\{0,\cdots,p-1\}$ からランダムに選ぶ。

Step 4: $y \equiv g^x \pmod{p}$ を求める。

以上のようにして求めた p, g, y を公開鍵として公開し、x を秘密鍵として本人だけが知るように保管する。

3.1.2 暗号化

送信者が受信者にメッセージ$m \in Z_p^*$を送るとする。送信者はまず、乱数 r を$\{0,\cdots,p-1\}$から定める。次に、受信者が公開している公開鍵 p, g, y を探し、公開鍵と r を用いて次のように暗号化を行う。

$$c_1 \equiv g^r \pmod{p}$$
$$c_2 \equiv my^r \pmod{p}$$

得られた暗号文(c_1, c_2)を受信者へ送る。

3.1.3 復号化

受け取った暗号文(c_1, c_2)を、自分で保管していた秘密鍵 x を用いて次のように復号化を行う。

$$m \equiv c_2 \times (c_1^x)^{-1} \pmod{n}$$

このようにしてメッセージ m を復元する。なぜこのように復号化が行えるかというと、まず暗号化のc_2の計算式における y に注目する。鍵の生成で y はg^x乗と合同なので、

$$c_2 \equiv m(g^x)^r \pmod{p}$$

となる。次に、累乗の部分に注目し式変形を行うと

$$c_2 \equiv m(g^r)^x \pmod{p}$$

となる。ここでg^rに注目すると、g^rは暗号化のc_1となるので

$$c_2 \equiv mc_1^x \pmod{p}$$

となる。次に、両辺にc_1^xの逆元を掛けると

$$c_2(c_1^x)^{-1} \equiv mc_1^x \times (c_1^x)^{-1} \pmod{p}$$

となる。ここで右辺に注目すると、ある数とその逆元との積は 1 なので

$$c_2(c_1^x)^{-1} \equiv m \pmod{p}$$

と m だけが残る。このようにして秘密鍵 x を用いて復号化が可能となる。これを基にして、(t, N)しきい値 ElGamal 暗号を構成する。

3.2 (t,N)しきい値 ElGamal 復号

前章で述べた(t, N)しきい値 RSA 復号と同様、(t,N)しきい値 ElGamal 暗号とはシャミアのしきい値法を利用することにより、秘密鍵を N 人で分割・管理し、t 人以上によりメッセージを復元する ElGamal 分散復号である。まず、秘密鍵の生成を行う。事前に参加者間で q|p−1 となる素数 p, q を共有しておく。

アルゴリズム 3.2.1 (t, N)しきい値 ElGamal 復号における鍵の生成

Step 1: 参加者 A_i は乱数 $a_{ij}\ (0 \le j \le t-1)$ を定め、t−1 次の多項式 $f_i(z)$

$$f_i(z) = \sum_{j=0}^{t-1} a_{ij} z^j \pmod{q}$$

を生成する。そして、分散情報として多項式上の点 s_{ij}

$$s_{ij} = f_i(j)$$

を計算し、他の参加者達へ送る。

Step 2: 分散情報を受け取った A_i は、部分秘密鍵 x_i

$$x_i = \sum_{A_j} s_{ji} \pmod{q}$$

を計算する。このとき、秘密鍵 x は

$$x = \sum_{A_i} a_{i0} \pmod{q}$$

である。

Step 1 では、各参加者がディーラとして秘密情報を分散させている。Step 2 では、自分に送られてきた分散情報から部分秘密鍵を作成している。これらを用いて通信を行う。

今、参加者 A_i は秘密鍵 x に対する部分秘密鍵 x_i と、グループ宛の暗号文 (c_1, c_2) を持っているとする。まず、自分が保管している部分秘密鍵を用いて部分復号情報 s_i

$$s_i = c_1^{x_i} \pmod{p}$$

を計算する。求めた s_i を公開することにより、各参加者は t 人分の参加者の部分復号情報を得ることができる。ここで、参加者 A_i はラグランジェ係数 λ_i

$$\lambda_i = \prod_{j \neq i} \frac{x_j}{x_j - x_i}$$

を用いて、

$$c_1^{\,x} = \prod_{i=1}^{t} s_i^{\lambda_i}$$

を計算する。次にc_2の逆元を求め、$c_1^{\,x}$との積を計算することでメッセージ m を復元することができる。以上が(t, N)しきい値 ElGamal 分散復号となる。

4. 終わりに

本稿では、公開鍵暗号である RSA 暗号[6], ElGamal 暗号[2]を基にして、複数の受信者から復号化を行う RSA 分散復号[5], ElGamal 分散復号[3]について述べた。これらの分散復号の構成においては、秘密分散法の一つであるシャミアのしきい値法[7]を利用している。しかし、シャミアのしきい値法を利用した RSA 分散復号を構成するためには、第2.6節で述べたように、いくつかの難問をクリアしなくてはならない。そこで、今後は、シャミアのしきい値法とは別の秘密分散法を利用する構成法が重要になってくる。その構成法の一つとして、中国人剰余定理を利用した秘密分散法(Asmuth-Bloom 法[1])や、この Asmuth-Bloom 法を利用した RSA 分散復号[4]がある。

参考文献

[1] C. Asmuth and J. Bloom: A modular approach to key safeguarding. *IEEE Trans. Informat. Theory*, vol. 29, no. 2, 1983, pp. 208－210.

[2] T. ElGamal: A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Informat. Theory*, vol. 31, 1985, pp. 469－472.

[3] R. Gennaro, S. Jarecki, H. Krawczyk and T. Rabin: Secure Distributed Key Generation for Discrete-Log Based Cryptosystems. *Proceedings of Eurocrypt'99, Lecture Note in Comp. Sci.*, vol. 1592, 1999, pp. 295－310.

[4] Kamer Kaya and Ali Aydın Selcuk: Threshold cryptography based on Asmuth-Bloom secret sharing. *Inform. Sci.*, vol. 177, no. 19, 2007, pp. 4148－4160.

[5] G. Qiu, H. Wang, S. Wei and G. Xiao: Information-theoretic secure verifiable secret sharing over RSA modulus. *Wuhan Univ. J. Nat. Sci.*, vol. 11, no. 6, 2006, pp. 1849—1852.

[6] R. L. Rivest, A. Shamir, and L. Adleman: A method for obtaining digital signatures and public key cryptosystems. *Communications of the ACM*, vol. 21, 1978, pp. 120—126.

[7] A. Shamir: How to share a secret. *Communications of the ACM*, vol. 22, no. 11, 1979, pp. 612—613.